

Examen SQL - Turno 1

30/09/2022

El examen deberá ser entregado en un archivo de texto (extensión .sql o notebook) de tal forma que las consultas implementadas puedan ser ejecutadas directamente.

La base de datos a usar es la chinook del laboratorio (puede usarse una notebook personal). La entrega se debe realizar sin excepción en el campus y en la tarea correspondiente antes de la hora de finalización.

La aprobación requiere de tener al menos 4 (cuatro) consultas correctamente realizadas. Cada consulta deberá resolverse en una única consulta. No usar select en el from salvo que no haya otra manera de resolver la consulta y en ese caso podría usar CTE.

1. Obtener las facturación anual por país.
2. Cuáles son los clientes que realizaron más compras.
3. Obtener el track de mayor duración para cada género indicando el título del Álbum al que pertenece.
4. Cuáles son las playlist más vendidas.
5. Listar playlists con tracks de más de tres artistas.
6. Realizar una consulta correlacionada que devuelva, si es que lo hubiera, todos los customers que tengan como representante de ventas un empleado de otra ciudad a la que pertenece el cliente. Es decir que cliente y vendedor son de distintas ciudades.

1. Obtener la facturación anual por país.

```
select i.billing_country, date_part('year',i.invoice_date),
sum(i.total)
from invoice i
GROUP BY i.billing_country, date_part('year',i.invoice_date)
```

2. Cuáles son los clientes que realizaron más compras

```
with count_invoice_by_customer as (
  select c.customer_id, count(i.invoice_id) as count_invoice
  from customer c
  left join invoice i on c.customer_id = i.customer_id
  GROUP BY c.customer_id
)

select cibc.customer_id
from count_invoice_by_customer cibc
where cibc.count_invoice = (select max(count_invoice) from
count_invoice_by_customer)
```

3. Obtener el track de mayor duración para cada género indicando el título del Álbum al que pertenece.

```
with time_duration_track_by_genre as (
  select t.genre_id, max(t.milliseconds) as duration_max
  from genre g
  left join track t on t.genre_id = g.genre_id
  group by t.genre_id
), track_max_duration_by_genre as (
  select t.track_id, tdbg.genre_id, tdbg.duration_max
  from time_duration_track_by_genre tdbg
  inner join track t on
  tdbg.duration_max = t.milliseconds and tdbg.genre_id =
t.genre_id
)

select al.title, tm.track_id, tm.genre_id, tm.duration_max
from track_max_duration_by_genre tm
inner join track t on t.track_id = tm.track_id
inner join album al on al.album_id = t.album_id
```

4. Cuáles son las playlist más vendidas

```
with playlist_invoice as (  
    select plt.playlist_id, il.invoice_id  
    from playlist_track plt  
    inner join track t on t.track_id = plt.track_id  
    inner join invoice_line il on il.track_id = t.track_id  
) , count_invoice_by_playlist as (  
    select pin.playlist_id, count(distinct pin.invoice_id) as  
count_invoice  
    from playlist_invoice pin  
    group by pin.playlist_id  
)  
  
select cibp.playlist_id  
from count_invoice_by_playlist cibp  
where cibp.count_invoice = (select max(count_invoice) from  
count_invoice_by_playlist)
```

5. Listar playlists con tracks de más de tres artistas.

```
with count_artist_by_playlist as (  
    select plt.playlist_id, count(DISTINCT al.artist_id) as  
count_artist  
    from playlist_track plt  
    inner join track t on t.track_id = plt.track_id  
    inner join album al on al.album_id = t.album_id  
    GROUP BY plt.playlist_id  
)  
  
select ca.playlist_id  
from count_artist_by_playlist ca  
where ca.count_artist > 3
```

6. Realizar una consulta correlacionada que devuelva, si es que lo hubiera, todos los customers que tengan como representante de ventas un empleado de otra ciudad a la que pertenece el cliente. Es decir que cliente y vendedor son de distintas ciudades.

```
select c.customer_id
from customer c
where EXISTS (
    select e.employee_id
    from employee e
    where
        c.support_rep_id = e.employee_id and
        e.city != c.city AND
        e.state != c.state
)
```